Petri Kettunen. 2007. Extending software project agility with new product development enterprise agility. Software Process: Improvement and Practice, volume 12, number 6, pages 541-548.

SOFTWARE PROCESS IMPROVEMENT AND PRACTICE
Softw. Process Improve. Pract. 2007; **12**: 541–548
Published online 25 June 2007 in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/spip.342

Extending Software Project Agility with New Product Development Enterprise Agility

Practice Section

Petri Kettunen*,†
Nokia Siemens Networks, P.O. Box 6, FI-02022 Nokia Siemens Networks, Finland

Many new product development (NPD) companies must nowadays work under turbulent conditions. The market competition is often fierce, while at the same time new and sometimes even disruptive technological uncertainties emerge. Speed and flexibility are then key success factors. Agility is thus a prospective strategy for such companies. The (embedded) software product development projects working in such environments face the turbulence either directly or indirectly, and agile software methods could potentially help to cope with the challenges. However, in large NPD environments the software project agility must be combined with the overall enterprise agility in order for the company to meet the ultimate business goals. This exploratory article investigates software project agility from that point of view by constructing a framework of the considerations for larger NPD companies to achieve more overarching agility. The software process improvement (SPI) activities can then be focused accordingly. An industrial product development case example is illustrated, suggesting how software product development agility should be improved within the larger organizational context (SPI-in-the-large). Copyright © 2007 John Wiley & Sons, Ltd.

KEY WORDS: software process improvement; agile software process models; new product development; agile enterprise

1. INTRODUCTION

In the 1990s, many agile software methodologies – such as XP, FDD, and Scrum – emerged. Their key goal is to attempt to maximize the customer value of the actually delivered software product. Project success is determined in terms of this current business value instead of the traditional plan-driven front-end requirements and budget conformance.

The basic premise of those agile software methodologies is that a small, co-located self-organizing team working closely together with the business customer can produce high-value, high-quality software efficiently with rapid iterations and frequent feedback. This applies to single-project contexts.

However, in larger new product development (NPD) organizational context the set-up is more complicated. Typically there is a portfolio of concurrent product development projects to be managed. Large organizations developing large, complex systems have to control not only individual projects and products but also their interplay and often long product life cycles and large, diverse customer bases. All this should nevertheless be

* Correspondence to: Petri Kettunen, Nokia Siemens Networks, P.O. Box 6, FI-02022 Nokia Siemens Networks, Finland
† E-mail: petri.kettunen@nsn.com

accomplished with appropriate speed and flexibility.

The problem is, then, how the agile software projects can be aligned so that the agility of the whole NPD company can be achieved. We must thus expand the perspective of agile software methodologies and – consequently – the software process improvement (SPI) activities from project level to enterprise level. This is the aim of the current article.

The rest of the article is organized as follows. Section 2 explores the background and related work, and sets the exact questions. Section 3 then proposes answers with an example case in Section 4, followed by discussion and implications in Section 5. Finally, Section 6 gives conclusions with pointers to further research.

2. BACKGROUND AND RELATED WORK

2.1. Agile Software Product Development

There is no one universal definition of agility in software production. The Agile Manifesto is quite broadly formulated. For example, Conboy and Fitzgerald (2004) consider it too informal to be a proper basis for systematic analysis, and formulate a more rigorous general definition of agility.

Currently, many different publicly documented and advocated agile software development methodologies are available. They can be compared and contrasted from many different points of view, such as the following:

- What is their level of concern (individual *vs* enterprise) (Boehm and Turner 2004)?
- What is their life cycle scope (Abrahamsson *et al*. 2002)?
- What project problems each method tackles (Kettunen and Laanti 2005)?
- What is their 'level of agility' (Schwaber 2001)?
- What are their value stream cost structures (Anderson 2004)?

Often no one agile software method is in practice a complete solution for all situations. Consequently, there is a trend to combine and adapt different methods and practices to various hybrids. In large established organizations, there may be additional limitations and constraints to be taken into account (Lindvall *et al*. 2004). Even cultural shifts may be needed (Nerur *et al*. 2005).

Since agile software methods are now being taken increasingly in use in larger organizations, the original assumptions of small customer-coupled project teams have to be stretched. There are several attempts to extend the current agile methodologies (McMahon 2005). Typically the approach is to combine multiple small agile software project teams into larger management entities (e.g. Scrum of Scrums).

Often the key issue in large organizations is to be able to overcome the formal organizational boundaries (even static 'silos') and/or geographical distribution, and bring the relevant cross-functional teams together into close co-operation with intense communication (Kähkönen 2004).

2.2. Agile Enterprises

In manufacturing sector, the concept of agility was recognized in early 1990s (Preiss 2005). The key idea is to create flexible manufacturing systems, capable of accommodating changes and unpredictability in product demand. In addition, agile supply chain management tunes the production stream under such conditions for innovative products (Lee 2002).

A notable principle in agile manufacturing and supply chain management is that the entire operation of the company production stream is directed towards those goals, thus avoiding isolated local optimizations. Furthermore, the business processes are intimately connected with the production processes.

The phase of the products evolution life cycle and ultimately, the current strategic direction of the whole company affect the needs and strategic choices of the enterprise agility (Kontio 2005). For example, does the company strive for global product leadership in certain customer segments, or customer intimacy with selected key customers (Treacy and Wiersema 1995)?

An organization-wide perspective is therefore important in order to take full advantage of agility in product development. A robust organization as a whole is able to sustain profitable business even under external turbulence (Conboy and Fitzgerald 2004). Response ability (i.e. the capability to confront and effectively deal with external and internal changes) is a key factor to this (Dove 2004). The software development projects are parts of this system.

Softw. Process Improve. Pract., 2007; **12**: 541–548
DOI: 10.1002/spip

The limitations of traditional project management methodologies in such environments have been recognized not only in software product development, but also in general. Agile project management is thus an emerging competence in agile enterprises (Thomsett 2002, Wysocki 2003, Chin 2004).

2.3. Agility and Software Process Improvement

In general, the purpose of SPI is to develop the software production capabilities of the organization. Agility can be seen as one element of those capabilities.

Traditionally, in large organizations with centralized process management, the SPI activities are typically driven by organizational goals (e.g. productivity). However, the agile software development models emphasize more local team-level SPI (Salo and Abrahamsson 2007).

With respect to large-scale NPD agility, the SPI activities can be scoped at different levels ranging from the team level to the enterprise level. Börjesson (2006) has demonstrated how large product development organizational SPI can be accomplished in agile ways.

2.4. Exploration Questions

The background line of reasoning leads to the following specific questions:

1. How does software project agility relate to NPD enterprise agility?
2. What are the implications for SPI?

In the following section, we tackle these questions with a constructive approach. Our primary focus is in large-scale (embedded) software product development.

3. COMBINING NPD ENTERPRISE AND SOFTWARE PROJECT AGILITY

3.1. Agile Software Product Development Projects in the NPD Enterprise Context

Figure 1 illustrates a simplified, ideal model of a customer-driven agile software development project set-up. The project team, appropriately resourced and empowered, works in close cooperation with the business customer. Frequent iteratively developed software product releases

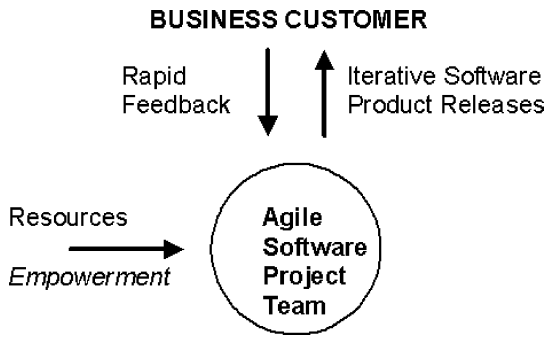

Figure 1. A basic model of agile software projects

follow immediate customer feedback, thus meeting the current customer needs as closely as possible, providing maximum value.

However, in larger market-driven product development contexts, the software project team set-up is in reality much more complicated like illustrated in Figure 2. In particular, the following considerations have to be taken into account:

- Who are the (main) customers? Where do they come from?
- Does the company want to satisfy all of them equally well, or focus on certain key customers/segments, for example, with product customization/varieties?
- In addition to the direct customer feedback, there are other business and technological environment inputs to be considered (e.g. competitors).

Comparing this to the simple case in Figure 1, we can see that a software project faces many additional complications:

- The project team may not have a direct connection to the business customer(s). In fact, it

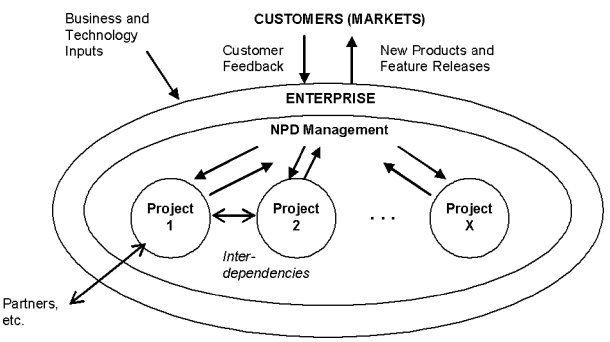

Figure 2. Modeling agile software projects in large NPD organization context

DOI: 10.1002/spip

may be difficult to identify any particular customer. Consequently, the product requirements and priorities are set by other stakeholders.
- The project team may have many interdependencies both internally (e.g. with the hardware development projects) and externally (e.g. subcontractors).
- The NPD management may not be able or willing to fully provide the needed resources and/or empowerment for all the projects in the multi-project portfolio. Furthermore, the situation may have to be readjusted during the course of the project work.

Figure 3 illustrates a practical example of those considerations with the Scrum method. The basic process model has to be connected with the rest of the NPD organization (*cf* Figure 2). Depending on the actual project context, this may involve issues like the following (see the numbered connector arrows in Figure 3):

- setting the project and product priorities (1 and 2 in Figure 3)
- systems engineering, possibly a legacy architecture (3)
- organizational process definitions (4)
- piloting (5)
- other products/components to be integrated (6)
- product releases delivery (7)
- internal documentation for the next release projects (8)

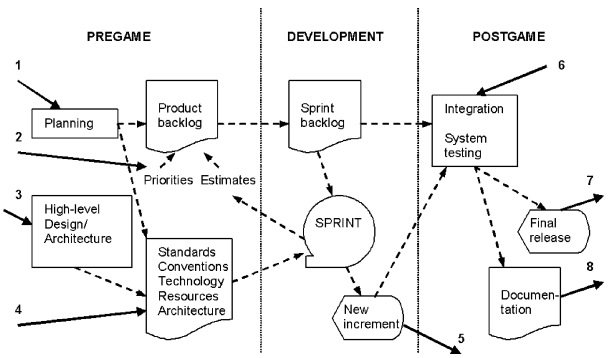

Figure 3. An example of an agile software method organizational extension (adapted from Abrahamsson *et al*. (2002))

3.2. NPD Software Project Agility Dimensions

Based on the reasoning at the organizational level presented above, it becomes more understandable how individual software projects fit into the larger organizational agility context (Question 1 in Section 2.4). Following that reasoning, it is possible to characterize the dimensions of software project agility in the NPD enterprise context. Figure 4 illustrates this. The more market and technology uncertainty there is (what product to build and how), the more flexible the product and design approaches should be to accommodate the volatility and changes. However, the project complexity and organizational constraints limit the choices, and bind the project management.

Different agile software methodologies address those dimensions to various degrees. The key is thus to understand and agree on the interfaces and constraining factors between the project team and the rest of the NPD organization. Some of the key considerations are then, for example, the following:

- What is the customer interface and the 'distance' between the customer(s) and the project team?
- Who makes the business decisions about the product features and the development schedule?
- What kind of decisions is the project team authorized to make?
- What is the required level of project progress visibility?
- What is the skill level of the available people?

3.3. NPD Enterprise Agility Space

Combining the enterprise model (Figure 2) with the software project agility model (Figure 4) produces a synthesis view of the overall NPD enterprise agility as shown in Figure 5 (Question 1 in Section 2.4). The

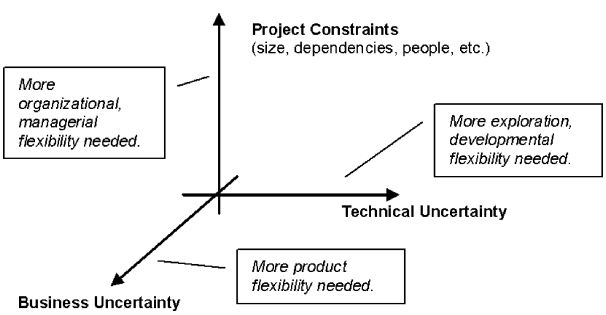

Figure 4. Project agility dimensions

Softw. Process Improve. Pract., 2007; **12**: 541–548
DOI: 10.1002/spip

agility of the NPD organization is the aggregate of the product development projects. Consequently, the different projects in the portfolio may have very different drivers for their agility within the NPD organization.

For example, the product development platform projects are typically long-term strategic investments with many dependencies, while the individual product (enhancement) release projects may have shorter term goals. The NPD function as a whole has to balance its agility with the different projects. This includes launching and terminating new projects.

Finally, at the enterprise level the entire NPD organization is just one part of the agility. Other means are for example the product placement, manufacturing, and pricing. Furthermore, the enterprise may decide to co-operate with other organizations and even with some competitors (virtual company).

4. EXAMPLE CASE

As of this writing, we are not ready to publish exact empirical data about our propositions presented in Section 3. However, in our experience, the following example scenario can be typical in large-scale industrial NPD environments.

A large-scale embedded software product development project is running to produce a selected set of new features for a large telecommunications network element product. The time-to-market span is about one year. In the midst of this development period, the product marketing recognizes a new prospective business opportunity, but it requires the support of some additional product features not included in the current development set. The original time-to-market schedule cannot be compromised, though.

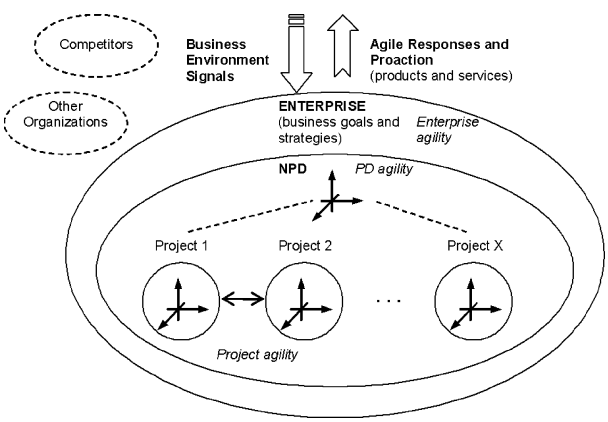

Figure 5. NPD enterprise agility model

In such a situation, different scenarios are possible, depending on the agility of the NPD organization and the software project management:

1. In a less agile NPD organization the product marketing and the R & D functions have only limited connections and collaboration. The product marketing offers the additional features without thoroughly negotiating with the R & D, which in turn does only limited impact analysis of the additional features. The software product development project follows an incremental delivery life cycle model, but the length of the increments is scheduled to be some six months. Consequently, the additional features are just added on top of the original work plan. However, because of the incomplete impact analysis, the additional features are later realized to be much more complicated than initially assumed, requiring considerable implementation work and extensive integration tests (feature interactions). The net result is that the software product development project runs into serious schedule trouble, and eventually fails to deliver not only the additional features but also the original content on time. The business impact as well as the internal satisfaction are negative.
2. A more agile NPD organization understands the need for intense and continuous co-operation between the product marketing and the embedded software development functions. The software development has realized the need to be reactive to external changes. The software product development project thus follows an iterative life cycle model with short (some one month) time-boxed iterations. The additional features are considered to be analysed for the next starting iteration. Since the product design analysis, done with the systems engineering, reveals the complexity of the new feature interactions with the current ones, some lower-priority features are renegotiated to be developed in a later additional iteration. Consequently, the original time-to-market schedule target can be met, albeit with some reduction of

DOI: 10.1002/spip

the original feature content, which will be delivered with the next iteration. The most important business targets are satisfied on time, and the product development work remains sustainable.

Reflecting this case with the constructs presented in Section 3, the key success point is to realize the positioning and the external connections of the software development projects in the enterprise NPD value stream (Figure 2). In the scenario 1, a less agile project management encounters problems trying to cope with the changes only reactively, whilst in the scenario 2 the more agile organization realizes the sources and nature of project uncertainty (Figure 4), and takes proactive measures accordingly. The software project's external connections are then managed systematically (Figure 3), and the changes can be implemented in a controlled, yet flexible way. The agile software development (project) makes it possible to realize flexible product development offerings, and consequently enable the overall NPD enterprise agility (Figure 5).

Note that this kind of a scenario is not unique to telecommunications product development. Similar concerns can be recognized for instance with industrial process control equipment products (Dagnino 2001).

5. ANALYSIS AND DISCUSSION

5.1. Relating Software Project Agility to NPD Enterprise Agility

In Section 2 we set a question of how to link agile software projects with the NPD enterprise agility (Question 1 in Section 2.4). In Section 3 we have constructed partial answers to this question.

Clearly, the propositions depend on the unique contextual factors of the particular company environment. Different companies make different strategic choices, for example, on the way they interface with the customers. Such company-specific choices influence profoundly the agility of the related software projects. It is therefore not reasonable to attempt to provide one-size-fits-all answers. However, we hypothesize the principles and patterns of our constructs presented in Section 3 to generalize to considerable extent in NPD companies. More empirical evidence is needed, though.

5.2. Implications for SPI

The implications of our propositions presented in Section 3 for SPI are as follows (Question 2 in Section 2.4). Here, we reflect the case example observations (Section 4).

The NPD company should first make clear strategic choices about how much agility and on which areas it strives for. This may then put totally different requirements for the different product development projects within the company. Notably the company as a whole can be agile externally, even though some of its internal software projects are not, because the agility stems from the combination of the individual projects. For example, there could be a long-term software product platform development following a plan-driven mode of operation, while there are short, flexible product derivative development projects for rapid responses to fast customer needs.

The agility of the individual software projects should not be developed in isolation, but the interface and organizational constraints must be taken into account (Figure 2). For example, the right choice of the appropriate agile software methods depends much on such overall factors. For some projects, even the traditional waterfall-based development may then be the best approach!

In turbulent business environments, agility is a dynamic operator. The choices both at the enterprise and project levels should be continuously reflected and, if necessary, revised according to the latest external market inputs. Developing strategically wrong (uncompetitive) products is wasteful, no matter how much agility there is otherwise in the product development (Rand and Eckfeldt 2004). In fact, a truly agile NPD enterprise avoids such failures, or is at least able to recover them quickly. Not all innovative products are successful.

Comprehensive management of NPD enterprise agility requires cross-functional understanding of not only software engineering management, but also such areas as industrial engineering and management, business economics, and supply chain management. Competence management with associated measurements is thus important for an agile organization (Banerjee and Bhattacharya 2002).

The company management systems and the distribution of the decision-making power with the relevant knowledge have to be aligned across the organization. Similar issues have been recognized

Softw. Process Improve. Pract., 2007; **12**: 541–548
DOI: 10.1002/spip

even in modern military organizations (Delegated Autonomous Command) (Atkinsson and Moffat 2005).

Large, established NPD organizations may find it more difficult to transform themselves into fully agile enterprises than smaller companies. This is, for example, because of the existing organizational boundaries, legacy products, and even the prevailing organizational culture.

Salo and Abrahamsson have recently developed an agile process improvement model focusing on the software project team (group) level (Salo and Abrahamsson 2007). However, they make a clear note that the team-level SPI may be constrained by the organizational level and management, and some improvement actions may require external changes beyond the project team control. A proficient facilitator, preferably outside the project team, is thus recommended. The organizational engagement and support of the relevant stakeholders are important.

6. CONCLUSIONS

In this article, we have investigated agile (embedded) software development projects and SPI in larger NPD enterprise context. If the company could fully exploit such options, the agile software projects would be naturally incorporated. In such an environment, they are appropriately resourced, empowered, and governed. A truly agile project team even attempts to influence and change the surrounding organization, if necessary, to reach the goals (Chin 2004/Ch. 6).

The software project teams in turn understand all the time, how they serve the company business best, even under turbulence. The project teams realize their positioning in the NPD organization, and what business effects they are expected to bring at the enterprise level. The software process models used are selected and adapted accordingly. The life cycle pacing of the different projects are aligned at the enterprise level for both short-term responsiveness as well as for longer term capability developments.

In this article, we have proposed certain frames for extending software project agility with NPD enterprise agility. A key conclusion is that the software project agility should be aligned with the overall NPD strategy, and – ultimately – with the enterprise agility. The SPI activities should be mapped to those enterprise-level influences (SPI-in-the-large) in order to avoid isolated optimizations.

Having stated that, this article leaves room for further study:

- Supporting our propositions with empirical evidence.
- There is a need to develop appropriate metrics for assessing the level of agility (or the lack of it) and the consequent business effects both at the project level as well as in the larger enterprise context.
- Developing our propositions further towards a more systematic framework for developing the agility in large NPD organizations, starting from the strategic business objectives. In particular, what enabling factors are needed, and what are the main obstacles – taking into account the actual organizational context? One proposition towards that goal is presented by Highsmith (Highsmith 2005). We have already addressed those questions partially elsewhere (Kettunen 2006, Kettunen and Laanti 2006).

So far, not many large (if any) industrial companies have been able to achieve this enterprise-wide agility to the full in practice. One can ask, how much potential there is for improving the agility of software product development, in general, in NPD enterprises by considering the software agility in the larger context. The potential benefits are remarkable, if the company is able to combine the entrepreneurial small-scale agility with large-scale production economics.

ACKNOWLEDGEMENTS

The author would like to thank Pekka Abrahamsson (VTT Technical Research Centre of Finland) for commenting an earlier version of this article.

REFERENCES

Abrahamsson P, Salo O, Ronkainen J, Warsta J. 2002. Agile software development methods. Review and analysis, http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf.

Anderson DJ. 2004. *Agile Management for Software Engineering*. Prentice Hall: Upper Saddle River, NJ.

Atkinsson SR, Moffat J. 2005. The Agile Organization: From Informal Networks to Complex Effects and Agility,

DOI: 10.1002/spip

http://www.dodccrp.org/publications/pdf/Atkinson_Agile.pdf.

Banerjee N, Bhattacharya S. 2002. Creating an agile software development organization: a key factor for survival in today's economy. *Proceedings of the International Engineering Management Conference (IEMC)*, Cambridge.

Boehm B, Turner R. 2004. *Balancing Agility and Discipline – A Guide for the Perplexed*. Addison-Wesley: Boston, MA.

Börjesson A. 2006. *Making Software Process Improvement Happen*. IT University of Gothenburg: Sweden.

Chin G. 2004. *Agile Project Management: How to Succeed in the Face of Changing Project Requirements*. AMACOM: New York.

Conboy K, Fitzgerald B. 2004. Toward a conceptual framework for agile methods: a study of agility in different disciplines. *Proceedings of the ACM Workshop on Interdisciplinary Software Engineering Research (WISER)*, Newport Beach, CA.

Dagnino A. 2001. Coordination of hardware manufacturing and software development life cycles for integrated systems development. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, Tucson, AZ.

Dove R. 2004. Enterprise agility – What is it and what fuels it? http://www.parshift.com/Essays/essay065.htm.

Highsmith J. 2005. Agile for the enterprise: From agile teams to agile organizations, http://www.cutter.com/project/fulltext/reports/2005/01/index.html.

Kähkönen T. 2004. Agile methods for large organizations – building communities of practice. *Proceedings of the Agile Development Conference (ADC)*, Salt Like City, Utah.

Kettunen P. 2006. Troubleshooting large-scale New Product Development embedded software projects. *Proceedings of the 7th International Conference on Product Focused Software Process Improvement (PROFES)*, Amsterdam, The Netherlands.

Kettunen P, Laanti M. 2005. How to steer an embedded software project: tactics for selecting agile software process models. *Proceedings of the International Conference on Agility (ICAM)*, Espoo, Finland.

Kettunen P, Laanti M. 2006. Combining agile software projects and large-scale organizational agility. *EuroSPI Industrial Proceedings*, Joensuu, Finland.

Kontio J. 2005. Why agile now? The agility needed for business success. *3rd Agile Software Development Seminar (ASDS)*, http://agile.vtt.fi.

Lee HL. 2002. Aligning supply chain strategies with product uncertainties. *California Management Review* **44**(3): 105–119.

Lindvall M, Muthig D, Dagnino A, Wallin C, Stupperich M, Keifer D, May J, Kähkönen T. 2004. Agile software development in large organizations. *IEEE Computer* **37**(12): 26–34.

McMahon P. 2005. Extending agile methods: a distributed project and organizational improvement perspective. *CrossTalk* **18**(5): 16–19.

Nerur S, Mahapatra R, Mangalaraj G. 2005. Challenges of migrating to agile methodologies. *Communications of the ACM* **48**(5): 73–78.

Preiss K. 2005. Agility – the origins, the vision and the reality. *Proceedings of the International Conference on Agility (ICAM)*, Espoo, Finland.

Rand C, Eckfeldt B. 2004. Aligning strategic planning with agile development: extending agile thinking to business improvement. *Proceedings of the Agile Development Conference (ADC)*, Salt Like City, Utah.

Salo O, Abrahamsson P. 2007. An iterative improvement process for agile software development. *Software Process: Improvement and Practice* **12**: 81–100.

Schwaber K. 2001. Will the real agile processes please stand up? http://www.cutter.com/project/fulltext/reports/2001/08/index.html.

Thomsett R. 2002. *Radical Project Management*. Prentice Hall: Upper Saddle River, NJ.

Treacy M, Wiersema F. 1995. *The Discipline of Market Leaders*. Addison-Wesley: Reading, MA.

Wysocki RK. 2003. *Effective Project Management: Traditional, Adaptive, Extreme*. Wiley Publishing: Indianapolis, IN.

DOI: 10.1002/spip